# CAST Engineering Dashboard - Metrics - Lines of code - Unexpected increase or decrease

**Description**

This page is a troubleshooting guide for issues related to number of lines of code.

**Observed in CAST AIP**

| Release | Yes/No |
|---|---|
| 8.3.x | ✓ |

**Observed on RDBMS**

| RDBMS | Yes/No |
|---|---|
| Oracle Server | ✓ |
| Microsoft SQL Server | ✓ |
| CSS | ✓ |

## Step by Step scenario

1. Generate a full snapshot(n-1) for first version of source code.
2. Generate a full snapshot(n) for second version of source code.
3. Compare the LOC for two versions and there is very huge decrease/increase in KLOC for snapshot n which is not expected.

# Action Plan

Perform the below actions:

1. If you are migrating and if you see an unexpected decrease or increase in LOC then check the section Unexpected increase or decrease in LOC after migration , if your case is not listed then go to next step.
2. If you are using or not using an extension then it will result in variation of LOC. For instance if you use an HTML5 extension in current version and were not using this extension in previous version analysis then it will cause an increase in KLOC. To check if you have used any extension run the below query on the KB containing the result of current and previous version analysis - CAST Universal Analyzer Framework - Information - How to find which version of COE UA Config pack has been installed
3. Calculate the LOC in KB for snapshot N-1 and Snapshot N

    a. If LOC of snapshotN and N-1 are equal then the problem is during compute snapshot
        i. Calculate the LOC in CB for all modules for Snapshot N and snapshot N-1
        ii. Identify the module with variation in number of objects
        iii. Check if the objects missing in Snapshot N or Snapshot N-1 are part of any module - SQL Queries - CAST Knowledge Base - Queries on Module - Module content and properties
            1. If they are part of any module then this is a bug contact CAST Technical Support with Relevant input
            2. If they are not part of the module then navigate to CMS Application - Modules

    b. If LOC of snapshotN and N-1 are not equal then this is a problem in analyzer -
        i. Look for any syntax errors (or some other message/warning/error for example : Table XXX is rejected because the first matching rule id 1 is ambiguous. This is useful info for Number of Table ) in the analysis log, which could explain why a lower number than expected was achieved. Please refer to CMS Snapshot Analysis - Run Analyzer - Non Fatal error  and CMS Snapshot Analysis - Run Analyzer - Warnings for further information.
        ii. Maybe the object was a ghost objet that is cleand up in the current run, check if the object was a ghost object in the previous run, for this refer to SQL Queries - CAST Knowledge Base - Queries on objects - Identify and clean up all the ghost objects and ghost projects in your knowledge base, if it's the case then it's expected to get the variation, the current LOC is the correct one.
        iii. Find the positions/bookmarks of the object - Check if the Start and End line is correct in KB
            1. If the Start and the End line of the object is incorrectly stored then you need to use a CAST AIP version > 7.3.x since this feature is enhanced in 7.3.x. If you are already using CAST AIP version > 7.3.x or if the Start and the End line of the object is correctly stored then please contact CAST Technical Support with relevant inputs. In addition to this please provide us with the screenshot of the enlighten showing the object properties (F12) and source code highlighted (F11)

> ⓘ If you are Comparing LoC of CAST with 3rd party applications then it is expected to get a difference as calculation of LoC by CAST AIP is based on various parameters (commented lines /blank lines /...) and this can vary when compared to other applications calculation of LoC.

# Relevant input

1. CAST Support Tool (CST) - alias Sherlock export - without source code for initial investigation. May need source code later.
2. Backup of Knowledge , management database and logs of previous version
3. Complete screenshot of dashboard with URL pointing to the LOC that you are looking at . If you are seeing the difference in the Version Comparison page, then you need to provide the comparison context i.e. corresponding system, application or module name
4. Expectations and justification for expectation

## Unexpected increase or decrease in LOC after migration

1. If you are migrating from 7.0.x (x<20) and 7.3.x, you may observe on the dashboard a change of LOC (Line of Code) for PL/SQL artifacts even though the source code has not changed. Between CAST AIP 7.0.x (x<20) and CAST AIP 7.3.x, numerous improvements have been made to the **offline PL/SQL Analyzer** related to Quality Rules that impact the way in which the LOC value is determined. For example, prior to CAST AIP 7.0.19, comments in routines defined in package headers and comments in some private routines defined in package bodies were not correctly detected as comments. This was corrected in 7.0.20. As a direct result of these improvements, when migrating from CAST AIP.0.x (x<20) to 7.3.x, and after running a snapshot in CAST AIP 7.3.1, results may show a change in the LOC value for a given artifact.
2. If you have an unexpected decrease in LOC after you migrate your .NET extension from a lower version of 1.0.x to 1.0.12 and above. Also from CAST AIP 8.3.16 onward there is a change in DMT where if two or more .csproj is pointing to the same source code only one .csproj will be kept on DMT level and rest will be removed. The reason why this has been done is because the analysis was hanging during the Comparing Object step due to the duplicate objects caused by two .csproj pointing to the same source code.
3. If you have an increase in LOC for js files after migration from 7.3.x to 8.x.x , then this is expected behavior as in 7.3.x there were some projects whose js files were not considered as internal because they were not in the scope of the analysis unit and hence considered as external which results in non-calculation of LOC for those js files. But from 8.x.x. since this is 64 bit and there is no limit of scope so all the js files are considered as internal to the project which results in an increase in LOC for js files.
4. If you have a decrease of LOC for ABAP after migration to 8.3.2 or newer, then this is expected. Starting from 8.3.2, the ABAP analyzer considers as external the XML files generated by SAP itself during extraction. You can check if a file is in this case by opening it and checking if the author is "SAP" like below :

   **SAP generated file**

   ```
   <PROGRAM
       name="LYCO2UXX"
       status="SYSTEM"
       description="Something"
       type="INCLUDE"
       creationdate="20021118"
       author="SAP*"
       lastmodificationdate="20021118"
       lastmodificationauthor="SAP*"
   />
   ```

## Calculate the LOC in KB

**PLEASE NOTE:  THE QUERIES BELOW ARE ONLY VALID IN CAST AIP 8.3.36 AND BELOW.  FOR CAST AIP 8.3.37 AND ABOVE YOU MUST MODIFY THE QUERIES AS FOLLOWS:**

- **change references:**
  - **properties = 0**
- **to**
  - **properties & 255 = 0**

Follow the below steps to calculate the Number of line of code stored in Knowledge Base. You have to run the set of queries twice. Once on the Knowledge Base containing analysis of VersionN and once on the Knowledge Base containing the analysis of Version N-1

1. Get the technical module ID by running the below query on the Knowledge Base

   ```
   select s.idroot as "Module ID" , dos.object_name as "Module Name"
   from setroot s, dss_objects dos
   where s.idset = dos.object_id
   and dos.object_type_id = 20000
   order by s.idroot
   ```

   Output -
   **389**;"My Application full content"

2. To get **the value of the Number of code lines** run the following query on the Knowledge Base. Note that only the application's non-external objects are considered.

```
SELECT SUM(oi.infval)
FROM   objinf oi,
       keys k
WHERE  oi.idobj IN (SELECT ( object_id )
                    FROM   ctt_object_applications
                    WHERE  properties = 0
                           AND application_id = <Technical_module_id>)
       AND inftyp = 1
       AND infsubtyp = 0
       AND k.idkey = oi.idobj
       AND k.objtyp IN (SELECT DISTINCT t.idtyp
                        FROM   typ t,
                               typcat tc
                        WHERE  t.idtyp = tc.idtyp
                               AND tc.idcatparent IN (SELECT cat.idcat
                                                      FROM   cat
                                                      WHERE
                                   catnam LIKE '%APM Sources%'))
```

**Example**

```
SELECT SUM(oi.infval)
FROM   objinf oi,
       keys k
WHERE  oi.idobj IN (SELECT ( object_id )
                    FROM   ctt_object_applications
                    WHERE  properties = 0
                           AND application_id = 389)
       AND inftyp = 1
       AND infsubtyp = 0
       AND k.idkey = oi.idobj
       AND k.objtyp IN (SELECT DISTINCT t.idtyp
                        FROM   typ t,
                               typcat tc
                        WHERE  t.idtyp = tc.idtyp
                               AND tc.idcatparent IN (SELECT cat.idcat
                                                      FROM   cat
                                                      WHERE
                                   catnam LIKE '%APM Sources%'))
```

Output - 2000

3. Get the list of objects used to compute the Number of Code Lines per module for all modules

```
SELECT k.*
FROM   objinf oi,
       keys k
WHERE  oi.idobj IN (SELECT ( object_id )
                    FROM   ctt_object_applications
                    WHERE  application_id = <application_id >
                           AND properties = 0)
       AND inftyp = 1
       AND infsubtyp = 0
       AND k.idkey = oi.idobj
       AND k.objtyp IN (SELECT DISTINCT t.idtyp
                        FROM   typ t,
                               typcat tc
                        WHERE  t.idtyp = tc.idtyp
                               AND tc.idcatparent IN (SELECT cat.idcat
                                                      FROM   cat
                                                      WHERE
                                   catnam LIKE 'APM Sources')
                       )
```

**Example**

```
SELECT k.*
FROM   objinf oi,
       keys k
WHERE  oi.idobj IN (SELECT ( object_id )
                    FROM   ctt_object_applications
                    WHERE  application_id = 389
                           AND properties = 0)
       AND inftyp = 1
       AND infsubtyp = 0
       AND k.idkey = oi.idobj
       AND k.objtyp IN (SELECT DISTINCT t.idtyp
                        FROM   typ t,
                               typcat tc
                        WHERE  t.idtyp = tc.idtyp
                               AND tc.idcatparent IN (SELECT cat.idcat
                                                      FROM   cat
                                                      WHERE
                                   catnam LIKE 'APM Sources')
                       )
```

Output -

"**idkey**";"**keynam**";"keylib";"keytyp";"keysubtyp";"keyclass";"keyprop";"objtyp";"idusrdevpro";"keydevdat";"keydevvlddat";"status";"sqlowner"
**462**;"**PG1004**";"";"XXXXXX";-1;14080;4;545;"???";"1970-01-01 00:00:00";"";0;""

## Calculate the LOC in CB

Run the below query twice for each module in SnapshptN and Snapshot N-1

```
select dos.object_name,dmr.metric_num_value from dss_objects dos join dss_metric_results dmr on dos.object_id
= dmr.object_id join dss_link_info dl on dl.next_object_id = dos.object_id where dmr.metric_id = 10151 and
dmr.snapshot_id = <Snapshot_id> and dl.previous_object_id = <Module_id> and dl.link_type_id = 3
```

Module_id can be fetched by using the page - SQL Queries - CAST Central Base - Queries on modules - How to get the ID of the functional module
Snapshot_id can be fetched by using the page - SQL Queries - CAST Central Base - Queries on snapshots - How to get the ID and name of a snapshot for a given application

## Find the positions/bookmarks of the object

Identify the positions/bookmarks by executing the below query on the KB -

```
select Info1 as "Start Line Number" , Info3 as "End Line Number"
from objpos
where idobj = <object_id>
```

The decrease in KLOC could be due to incorrect storage of the start & end positions of the object due to which there may be overlapping and loss of KLOC.To confirm the same you can check the Start and End position of the object by looking directly into the source file.

To verify this you may also connect to enlighten, drag the object on the view and hit the F11 key to view its source code and the F12 key to view its properties. This will show the number of lines of code for that object and the actual source code highlighted.

## The difference in LOC is seen at Module Level

If the difference in LOC is seen at Module Level then you need to drilldown to module level & investigate per module as shown below -

1. Follow the Steps given under Calculate the LOC in KB in the page - CAST Engineering Dashboard - Metrics - Lines of Code - Incorrect value for the first time on the KB containing analysis results of current version
2. Repeat it for the second time on the KB containing analysis results of previous version.
3. You will get the number of objects contributing to LOC in each module.
4. If there are less / more number of objects contributing in the current version then it explains the reason for the difference in LOC.
5. If you are unable to come to a conclusion after following the above steps then please contact CAST Technical Support with relevant inputs

**Related Pages**

**Notes/comments**

Ticket # 9304